On Message-Passing, MAP Estimation in Graphical Models and DCOPs

Akshat Kumar, William Yeoh, and Shlomo Zilberstein

Computer Science Department
University of Massachusetts
Amherst, MA 01003
{akshat,wyeoh,shlomo}@cs.umass.edu

Abstract. The maximum a posteriori (MAP) estimation problem in graphical models is a problem common in many applications such as computer vision and bioinformatics. For example, they are used to identify the most likely orientation of proteins in protein design problems. As such, researchers in the machine learning community have developed a variety of approximate algorithms to solve them. On the other hand, distributed constraint optimization problems (DCOPs) are well-suited for modeling many multi-agent coordination problems such as the coordination of sensors in a network and the coordination of power plants. In this paper, we show that MAP estimation problems and DCOPs bear strong similarities and, as such, some approximate MAP algorithms such as iterative message passing algorithms can be easily tailored to solve DCOPs as well.

Keywords: Graphical Models, MAP Estimation Problems, DCOPs

1 Introduction

Markov random fields (MRFs) [32] is a graphical model, where nodes in a graph correspond to random variables and edges in a graph correspond to potential functions between pairs of random variables. A common problem in MRFs is to compute the maximum a posteriori (MAP) assignment, which is the most probable assignment to all the random variables of the underlying graph. The MAP estimation problem is a problem common in many applications such as computer vision, bioinformatics and information theory. For example, they are used to identify the most likely orientation of proteins in protein design problems [35; 25]. As such, researchers, who are typically from the machine learning community, have developed a variety of approximate inference algorithms [32; 24] to solve them.

Distributed constraint optimization problem (DCOP) [15; 18; 38] is also a graphical model; nodes in a graph correspond to agents, where each agent can take on a value, and edges in a graph correspond to constraint functions between pairs of agents, where each constraint function is a function of the values of a pair of agents. The agents in a DCOP coordinate with each other via messages

to optimize all their constraint functions. DCOPs have emerged as a popular model for representing multi-agent coordination problems where the primary interactions are between subsets of agents. Examples of such problems include the scheduling of meetings [13], the coordination of sensors in networks [3], the management of power plants [11] and the generation of coalition structures [27]. As such, researchers, who are typically from the multi-agent systems community, have developed a variety of optimal DCOP algorithms [14; 15; 18] and approximate DCOP algorithms [4; 6; 16] to solve them.

Since both MAP estimation problems and DCOPs are graphical models, they share many similarities. The main difference is MAP estimation problems are centralized problems while DCOPs are decentralized problems. However, many (centralized) algorithms used to solve MAP estimation problems can be executed in a decentralized fashion and they can thus be tailored to solve DCOPs. For example, the max-sum algorithm [3] and algorithms using the Divide-and-Coordinate approach [28] that are used to solve DCOPs are motivated by MAP estimation algorithms [17; 10]. In this paper, we aim to highlight the similarities between the MAP estimation problem and DCOPs more explicitly, and show how a class of approximate MAP algorithms, namely iterative message passing algorithms, can be tailored to solve DCOPs. To the best of our knowledge, the connection between MAP estimation problems and DCOPs have not been explicitly made. Thus, it is our hope that this work will better bridge the two research communities, namely machine learning and multi-agent systems, and will help cross-fertilize them.

This paper is organized as follows. In Section 2, we describe MAP estimation problems, DCOPs and their similarities. In Sections 3 and 4, we give a brief overview of several iterative message passing algorithms and how they can be applied to solve DCOPs, and in Section 5, we describe their properties and space complexities. Lastly, we present our experimental results and conclusions.

2 Graphical Models

Probabilistic graphical models provide an effective framework for compactly representing probability distributions over high dimensional spaces and performing complex inference using simple local update procedures. In this work, we relate two optimization problems represented as graphical models: the *maximum a posteriori* (MAP) estimation in *Markov random fields* (MRFs) [32] and *distributed constraint optimization problems* (DCOPs) [15; 18; 38]. MAP estimation is crucial for many practical applications in computer vision and bioinformatics such as protein design [35; 25]. Computing the MAP exactly is NP-hard for general graphs [2]. Thus, approximate inference algorithms are often used [32; 24]. In this section, we will provide an overview of MAP estimation in MRFs and how they relate to DCOPs

2.1 MRFs and MAP Estimation Problems

A pairwise Markov random field (MRF) can be visualized by an undirected graph $G = (V, E)$. It is formally defined by

- A set of random variables $\mathbf{X} = \{x_i \mid \forall i \in V\}$, where each random variable has a finite domain of possible values that it can be assigned. Each random variable x_i is associated with node $i \in V$.
- A set of potential functions $\boldsymbol{\theta} = \{\theta_{ij}(x_i, x_j) \mid \forall (i,j) \in E\}$. Each potential function $\theta_{ij}(x_i, x_j)$ is associated with edge $(i,j) \in E$.

The complete assignment $\boldsymbol{x}$ to all the random variables has the probability:

$$p(\boldsymbol{x}; \boldsymbol{\theta}) \propto \exp\left(\sum_{ij \in E} \theta_{ij}(x_i, x_j)\right) \tag{1}$$

The objective of a maximum a posteriori (MAP) problem is to find the most probable assignment to all the variables under $p(\boldsymbol{x}; \boldsymbol{\theta})$. This objective is equivalent to finding a complete assignment $\boldsymbol{x}$ that maximizes the function:

$$f(\boldsymbol{x}; \boldsymbol{\theta}) = \sum_{ij \in E} \theta_{ij}(x_i, x_j) \tag{2}$$

Additionally, we assume without loss of generality that each θ_{ij} is non-negative in this paper. Otherwise, a constant can be added to each θ_{ij} without changing the optimal solution.

2.2 DCOPs

Like MRFs, a distributed constraint optimization problem (DCOP) with binary constraints can also be visualized by an undirected graph $G = (V, E)$, commonly called a constraint graph. It is formally defined by

- A set of agents $\mathbf{X} = \{x_i \mid \forall i \in V\}$, where each agent has a finite domain of possible values that it can take on. Each agent x_i is associated with node $i \in V$.
- A set of constraint functions $\boldsymbol{\theta} = \{\theta_{ij}(x_i, x_j) \mid \forall (i,j) \in E\}$. Each constraint $\theta_{ij}(x_i, x_j)$ is associated with edge $(i,j) \in E$.[1]

Therefore, agents and constraint functions in a DCOP correspond to random variables and potential functions in an MRF, respectively. Similar to the MAP estimation problem, the objective in a DCOP is to find the complete assignment

[1] Although the typical notation of a constraint function is F_{ij} or c_{ij} in the DCOP literature, we use the notation in the machine learning literature to better illustrate the mapping between MAP estimation problems and DCOPs.

$\boldsymbol{x}$ that maximizes the function $f(\boldsymbol{x};\boldsymbol{\theta})$ in Equation (2). The main difference between MAP estimation problems and DCOPs is that the former are centralized problems while the latter are decentralized problems. In MAP estimation problems, a single agent has complete knowledge of all potential functions and controls the value assignments of all random variables, while in DCOPs, each agent has knowledge of the constraint functions that it is involved in only and chooses its own value only. Nonetheless, many (centralized) MAP estimation algorithms can be executed in a decentralized fashion and can thus can be tailored to solve DCOPs.

3 Variational MAP Formulations and Algorithms

In this section, we provide an overview of two common variational formulation of the MAP estimation problem: *linear programming* (LP) and *quadratic programming* (QP) formulations. Most of the existing algorithms in the machine learning literature can be classified as solving either one of these two formulations. We then describe two such algorithms, namely the max-product linear programming (MPLP) algorithm of [5], which operates on the LP formulation, and the expectation-maximization (EM) algorithm of [12], which operates on the QP formulation.

3.1 Linear Programming Formulation

We now describe the first common variational formulation of the MAP estimation problem: a linear programming (LP) formulation. We first briefly describe the concept of marginal polytope that is often associated with the MAP estimation problem. The reader is referred to [32] for more details.

Let $\boldsymbol{\mu}$ denote a vector of marginal probabilities (also called mean parameters) for each node and edge of the MRF. That is, it includes $\mu_i(x_i)\ \forall i \in V$ and $\mu_{ij}(x_i, x_j)\ \forall (i,j) \in E$. The set of $\boldsymbol{\mu}$ that arises from some joint distribution p over all the variables of the MRF is referred to as the marginal polytope $\mathcal{M}(G)$:

$$\mathcal{M}(G) = \{\boldsymbol{\mu} \mid \exists p(\boldsymbol{x})\ s.t.\ p(x_i, x_j) = \mu_{ij}(x_i, x_j),\ p(x_i) = \mu_i(x_i)\} \tag{3}$$

The MAP estimation problem is then equivalent to solving the following LP:

$$\max_{\boldsymbol{x}} f(\boldsymbol{x};\boldsymbol{\theta}) = \max_{\boldsymbol{\mu}\in\mathcal{M}(G)} \boldsymbol{\mu}\cdot\boldsymbol{\theta} = \max_{\boldsymbol{\mu}\in\mathcal{M}(G)} \sum_{ij\in E}\sum_{x_i x_j} \mu_{ij}(x_i, x_j)\theta_{ij}(x_i, x_j) \tag{4}$$

Notice that p is a joint distribution over all the variables of the MRF – $p(x_1, \ldots, x_n)$ – and is, in general, very hard to represent and reason with. It can be shown that there always exists a maximizing solution $\boldsymbol{\mu}$ for the above problem that is integral and gives the optimal assignment $\boldsymbol{x}$. Unfortunately, the number of constraints used to describe this polytope are exponential and, thus, it can

not be solved efficiently. To remedy this problem, researchers have proposed LP relaxations that outer bound the polytope $\mathcal{M}(G)$. That is, the relaxed polytope $\mathcal{M}_L(G)$ is a super set of $\mathcal{M}(G)$ and may include certain fractional assignment to the random variables. This relaxation weakens the global constraint that $\boldsymbol{\mu}$ arises from some common distribution p. Instead, only pairwise (corresponding to the edges) and singleton consistency is required for mean parameters as given by the following condition:

$$\sum_{x_i} \mu_i(x_i) = 1 \qquad \forall i \in V \tag{5}$$

$$\sum_{\hat{x}_i} \mu_{ij}(\hat{x}_i, x_j) = \mu_j(x_j) \qquad \forall x_j, (i,j) \in E \tag{6}$$

$$\sum_{\hat{x}_j} \mu_{ij}(x_i, \hat{x}_j) = \mu_i(x_i) \qquad \forall x_i, (i,j) \in E \tag{7}$$

The constraints $\sum_{\hat{x}_i} \mu_{ij}(\hat{x}_i, x_j) = \mu_j(x_j)$ and $\sum_{\hat{x}_j} \mu_{ij}(x_i, \hat{x}_j) = \mu_i(x_i)$ ensure that, for each edge $(i,j) \in E$, the probability distribution μ_{ij} of the edge is consistent with the probability distributions μ_j of node $j \in V$ and μ_i of node $i \in V$, respectively. It might appear we can easily solve the LP of Eq. (4) subject to the above constraints, which are polynomial in the number of edges and domain size. However, even for moderately sized graphs, this LP becomes quite large with many constraints and the black-box LP solvers such as CPLEX do not scale well [35]. Therefore such LP is solved using specialized message-passing algorithms that take into account the graph structure of this LP [21; 25; 23]. There are several other algorithms which either work on this LP formulation or its dual such as the tree-reweighted max-product (TRMP) algorithm [30], convergent tree-reweighted algorithm (TRW-S) [9], the max-sum diffusion algorithm (MSD) [33; 34] and Lagrangian relaxation based approaches [7] among others.

3.2 Quadratic Programming Formulation

We now describe the second variational formulation of the MAP estimation problem: a quadratic programming (QP) formulation. The reader is referred to [22] for more details. Instead of Equation (4), the MAP estimation problem can also be formulated as a QP:

$$\begin{aligned} \max_{\mu_1,\ldots,\mu_n} \quad & \sum_{ij \in E} \sum_{x_i, x_j} \mu_i(x_i)\mu_j(x_j)\theta_{ij}(x_i, x_j) \\ \text{subject to} \quad & \sum_{x_i} \mu_i(x_i) = 1, \mu_i(x_i) \geq 0 \;\; \forall i \in V \end{aligned} \tag{8}$$

The above QP is compact even for large graphical models and has simple linear constraints – $O(|V|k)$ variables, where k is the maximum domain size, and $|V|$

Algorithm 1: Max-Product Linear Programming (MPLP) Algorithm

input: Graph $G = (V, E)$ and potential functions $\boldsymbol{\theta}$
output: Return complete assignment $\boldsymbol{x}$ s.t. $x_i = \text{argmax}_{\hat{x}_i}\, b_i(\hat{x}_i)$
repeat
  foreach *node* $i \in V$ **do**
    Send message $\gamma_{i \to j}$ to each neighbor $j \in Ne(i)$:
    $\gamma_{i \to j}(x_j) \leftarrow \max_{x_i} \left[\theta_{ij}(x_i, x_j) - \gamma_{j \to i}(x_i) + \frac{2}{|Ne(i)|+1} \sum_{k \in Ne(i)} \gamma_{k \to i}(x_i)\right]$
    Set node belief $b_i(x_i)$ to the sum of incoming messages:
    $b_i(x_i) \leftarrow \sum_{k \in Ne(i)} \gamma_{k \to i}(x_i)$
until *desired number of iterations*

Algorithm 2: Expectation Maximization (EM) Algorithm

input: Graph $G = (V, E)$ and potential functions $\boldsymbol{\theta}$
output: Return complete assignment $\boldsymbol{x}$ s.t. $x_i = \text{argmax}_{\hat{x}_i}\, \mu_i(\hat{x}_i)$
repeat
  foreach *node* $i \in V$ **do**
    Send message $\delta_{i \to j}$ to each neighbor $j \in Ne(i)$:
    $\delta_{i \to j}(x_j) \leftarrow \sum_{x_i} \mu_i(x_i)\ \theta_{ij}(x_i, x_j)$
    Set new marginal probability to sum of incoming messages:
    $\mu_i(x_i) \leftarrow \mu_i(x_i) \frac{\sum_{k \in Ne(i)} \delta_{k \to i}(x_i)}{C_i}$
until *desired number of iterations*

normalization constraints. Ravikumar and Lafferty also show that this formulation is *exact* [22]. That is, the global optimum of the above QP will maximize the function $f(\boldsymbol{x}; \boldsymbol{\theta})$ and an integral MAP assignment can be extracted from it. However, this formulation is non-convex and thus makes the global optimization hard. Nonetheless, the local optima of this QP correspond to good solutions for most problems empirically [12].

Next we describe two algorithms, the max-product LP (MPLP) [5] and the expectation-maximzation (EM) [12] which work on these two formulations. Both of these algorithms are based on iterative message-passing among neighboring agents and can be easily adapted for DCOPs.

3.3 MPLP and EM

Algorithms 1 and 2 show the pseudocode of the MPLP and EM algorithm, respectively. The MPLP algorithm operates on the dual of the LP formulation described in Section 3.1 and the EM algorithm operates on the QP formulation described in Section 3.2. In each iteration, every node i sends a message to each of its neighbor $j \in Ne(i)$. The messages in MPLP are denoted by $\gamma_{i \to j}$ and the messages in EM are denoted by $\delta_{i \to j}$. The marginal probability $\mu_i(x_i)$ in Algorithm 2 is the same as in the QP and the variable C_i is the normalization constant such that the resulting marginal probabilities sum up to one. Once the

Algorithm 3: Max-Sum (MS) Algorithm

input: Graph $G = (V, E)$ and potential functions $\boldsymbol{\theta}$
output: Return complete assignment $\boldsymbol{x}$ s.t. $x_i = \text{argmax}_{\hat{x}_i}\, b_i(\hat{x}_i)$
repeat
  foreach *node* $i \in V$ **do**
    Send message $m_{i \to j}$ to each neighbor $j \in Ne(i)$:
    $m_{i \to j}(x_j) \leftarrow \max_{x_i} \left[\theta_{ij}(x_i, x_j) + \sum_{k \in Ne(i) \setminus j} m_{k \to i}(x_i)\right]$
    Set node belief $b_i(x_i)$ to the sum of incoming messages:
    $b_i(x_i) \leftarrow \sum_{k \in Ne(i)} m_{k \to i}(x_i)$
until *desired number of iterations*

algorithm is run for a desired number of iterations, the complete assignment $\boldsymbol{x}$ can be extracted using the individual maximizer of the beliefs b_i for MPLP and marginal probabilities μ_i for EM of each node i. Alternatively, one can also run a better scheme to extract the solution of EM by running an additional iteration of message passing among neighbors [22]. It is quite clear that these algorithms can be easily adapted to solve DCOPs by letting each agent control its corresponding random variable and manage the messages that it sends and receives. Furthermore, the messages are computed based only on the shared information among immediate neighbors in the graph. Therefore, this property follows the DCOP specification that each agent knows only about the constraint functions with its immediate neighbors.

4 Reparameterization View of the MAP Problem

Since the max-sum (MS) algorithm [17] has been very successful in several contexts such as information theory [1; 31] and DCOPs [3], we provide an overview of some of the recently developed theoretical results behind it. Unlike the MPLP and EM algorithms, max-sum reparameterizes the original probability distribution of Equation (1) in terms of max-marginals for acyclic graphs and *pseudo*-max-marginals for cyclic graphs before estimating the MAP from these (pseudo)-max-marginals.

Algorithm 3 shows the pseudocode of the MS algorithm. Like the MPLP and EM algorithms, in each iteration, every node i sends a message $m_{i \to j}$ to each of its neighbor $j \in Ne(i)$. Once the algorithm is run for a desired number of iterations, the complete assignment $\boldsymbol{x}$ can be extracted using the individual maximizer of the beliefs b_i of each node i. Similar to the MPLP and EM algorithms, it is quite clear that this algorithm can also be easily adapted to solve DCOPs in the same way, as is already demonstrated [3].

We now briefly describe a reparameterization of the MAP estimation problem. The reader is referred to [31] for more details. Instead of max-sum, we discuss the max-product algorithm – max-sum messages can be considered as the log of messages in max-product which changes products to sums – which is more commonly described in the literature [31]. The max-product algorithm

maximizes the objective function of Equation (1) instead of the one of Equation (2). For comparison, their equivalent messages are:

$$m_{i\to j}^{sum}(x_j) \leftarrow \max_{x_i}\left(\theta_{ij}(x_i,x_j) + \sum_{k\in Ne(i)\setminus j} m_{k\to j}(x_i)\right) \tag{9}$$

$$m_{i\to j}^{prod}(x_j) \leftarrow \max_{x_i}\left(\exp\left(\theta_{ij}(x_i,x_j)\right) \prod_{k\in Ne(i)\setminus j} m_{k\to j}(x_i)\right) \tag{10}$$

where the former are the messages of max-sum and the latter are the messages of max-product. Assume that the algorithm converges and $m^\star$ are the messages upon convergence. We then define functions $T_i^\star$ for each node i and $T_{ij}^\star$ for each edge (i,j) as:

$$T_i^\star(x_i) = \prod_{j\in Ne(i)} m_{j\to i}^\star(x_i) \tag{11}$$

$$T_{ij}^\star(x_i,x_j) = \exp(\theta_{ij}(x_i,x_j)) \prod_{k\in Ne(i)\setminus j} m_{k\to i}^\star(x_i) \prod_{l\in Ne(j)\setminus i} m_{l\to j}^\star(x_j) \tag{12}$$

These functions $\mathbf{T}^\star$ define an alternative parameterization of the distribution $p(\boldsymbol{x};\theta)$ of Equation (1) as follows for both cyclic and acyclic graphs:

$$p(\boldsymbol{x};\mathbf{T}^\star) \propto \prod_{i\in V} T_i^\star(x_i) \prod_{(i,j)\in E} \frac{T_{ij}^\star(x_i,x_j)}{T_i^\star(x_i)T_j^\star(x_j)} \tag{13}$$

The following two properties hold for functions $\mathbf{T}^\star$:

- If the graph is acyclic, then the functions T_i and T_{ij} are equivalent to the max-marginals P_i and P_{ij}, which are defined as follows:

$$P_i(x_i) = \kappa \max_{\{\boldsymbol{x'}|x_i'=x_i\}} p(\boldsymbol{x'};\theta) \tag{14}$$

$$P_{ij}(x_i,x_j) = \kappa \max_{\{\boldsymbol{x'}|(x_i',x_j')=(x_i,x_j)\}} p(\boldsymbol{x'};\theta) \tag{15}$$

  where κ is a constant that can be different for each node i and edge (i,j). Intuitively, $P_i(x_i)$ is the probability of a most likely assignment $\boldsymbol{x'}$ with x_i' fixed to x_i in $\boldsymbol{x}$ and $P_{ij}(x_i,x_j)$ is the probability of a most likely assignment with x_i' and x_j' fixed to x_i and x_j in $\boldsymbol{x}$ respectively. Such reparameterization is helpful as the complete MAP assignment $\boldsymbol{x}^\star$ can then be extracted easily using one iteration of message-passing based on these max-marginals [31].
- If the graph is cyclic, then the functions $\mathbf{T}^\star$ do not represent the true max-marginals. However, they satisfy certain local consistency requirements, and

they are commonly referred to as the *pseudo*-max-marginals. The approximate MAP assignment for such graphs can then be extracted from such pseudo-max-marginals and, as empirically demonstrated [1; 31], is quite close to the optimal MAP assignment.

Wainwright and Jordan also show that any positive distribution $p(\boldsymbol{x})$ defined on a pairwise graph can be reparameterized in terms of pseudo-max-marginals [31]. Therefore, the fixed point of max-product updates exists for any arbitrary graph. Previously, this result was known only for acyclic graphs or graphs with only a certain number of cycles. This fundamental insight about the fixed point of max-product has resulted in the development of several successful and convergent message passing algorithms such as tree-reweighted max-product (TRMP) [30] and its variants [9].

5 Properties and Complexities of the Message Passing Algorithms

In this section, we discuss some of the desirable properties of approximate DCOP algorithms and specify whether the abovementioned algorithms have these properties.

5.1 Properties

- **Convergence:** An algorithm is said to have this property if it is guaranteed to converge to a fixed point after a *finite* number of iterations. For message passing algorithms, the fixed point is the point where the content of every message no longer changes. This property can be used as a basis to guarantee termination. The MPLP and EM algorithms have this property. The MS algorithm does not have this property unless it is operating on a acyclic graph. Most existing approximate DCOP algorithms like the k-, t- and C-optimal algorithms and those based on the Anytime Local Search framework have this property [16; 8; 29; 39].
- **Anytime:** An algorithm is said to have this property if it finds solutions whose qualities are monotonically non-decreasing. The EM algorithm has this property. Both the MPLP and MS algorithms do not have this property. Most existing approximate DCOP algorithms like the k-, t- and C-optimal algorithms and those based on the Anytime Local Search framework have this property [16; 8; 29; 39].
- **Error Bounded:** An algorithm is said to have this property if it has an error bound on the solution quality. The MPLP algorithm has this property. The upper bound UB can be calculated by using the γ messages as follows:

$$UB = \sum_{i \in V} \max_{x_i} \sum_{k \in Ne(i)} \gamma_{k \to i}(x_i) \tag{16}$$

Instance	Time (sec.)		Cycles		Solution Quality		Error Bound
	MPLP	BnB-Adopt	MPLP	BnB-Adopt	MPLP	BnB-Adopt	MPLP
1	0.181	496.75	403	55940	2402	2402	0.83
2	0.183	584.59	308	65646	3631	3631	0.20
3	0.807	470.75	466	52947	2718	2718	0.30
4	0.236	502.00	467	56613	2615	2615	0.47
5	0.160	484.32	382	54523	2758	2758	0.75

Table 1. Results for 4×4 grids

Furthermore, this upper bound is also monotonically non-increasing. The EM and MS algorithms do not have this property. Most existing approximate DCOP algorithms like the k-, t- and C-optimal algorithms have this property but, unlike the MPLP upper bounds, their upper bounds are determined a priori before the start of the algorithm [16; 8; 29].

– **Dynamic:** An algorithm is said to have this property if it is able to solve dynamic DCOPs, that is, DCOPs that change over time. The MPLP, EM and MS algorithms have this property. These message passing algorithms can handle dynamic DCOPs easily since they can compute the content of the messages and update their marginal probabilities/beliefs in the exact same way whether a DCOP changes or not. On the contrary, other pseudotree-based DCOP algorithms like ADOPT and DPOP need to reconstruct their pseudotrees for the new problem before solving it. However, it is important to note that there are several extensions of these pseudotree-based algorithms to handle dynamic DCOPs as well [19; 20; 26; 37].

5.2 Complexity Analysis

In this section, we describe the space complexity of the algorithms and their messages. In each iteration, the MPLP, EM and MS algorithms send $|Ne(i)|$ number of messages for each node $i \in V$, resulting in a total of $2|E|$ number of messages. Each of these messages contain k_i or, more generally, $O(k)$ floating point numbers, where k_i is the domain size of the sending agent x_i and k is the maximum domain size. Therefore, the network load in each iteration is $O(|E|k)$.

Each agent in all three algorithms needs to store the contents of the messages received in the previous iteration, the contents of the messages to be sent out in the current iteration and the marginal probabilities/beliefs. Therefore, the memory requirement of each agent is $O(2|Ne(i)|k + k) = O(|V|k)$.

6 Experimental Results

We now report some preliminary experimental results that we have obtained. We evaluated MPLP in sensor network problems using the PEAV formulation [13; 36]. The targets are arranged in a grid and each target is surrounded by four sensors, all of which are needed to track the target. We used two problem sizes: problems with 16 targets arranged in a 4×4 grid and problems with 25 targets

Instance	Time (sec.)	Cycles	Solution Quality	Error Bound
1	0.502	469	4773	0.06
2	0.685	697	5034	0.16
3	0.807	1087	5025	0.00
4	0.477	491	5279	1.13
5	0.433	447	5078	0.05

Table 2. Results for 5×5 grids

arranged in a 5×5 grid. The constraint costs are generated randomly. We also ran BnB-Adopt [36], an optimal DCOP search algorithm, as a baseline comparison.

We conducted our experiments on a machine with 8GB of RAM and 2.66GHz CPU. We measured the runtimes of the algorithms in the number of seconds and the number of (synchronous) cycles [15]. Table 1 shows the results for the 4×4 grids with 16 targets. The table shows that MPLP terminates significantly faster, in both seconds and cycles, compared to BnB-Adopt and still yields an optimal solution. The last column indicates the difference between the lower bound on the cost and the actual cost. Table 2 shows the results of MPLP for the 5×5 grids with 25 targets. Again it shows that MPLP finds near optimal solutions and the error bound is very small. We did not report the results for BnB-Adopt because the algorithm failed to terminate within a time limit of 2 hours.

The tables show that MPLP can find solutions with very small error bounds for both problem sizes and finds them by several orders of magnitude faster than BnB-Adopt. Although these results are preliminary, we believe that they demonstrate the potential of message passing algorithms to solve DCOPs.

7 Conclusions

Researchers in the machine learning community have long studied the maximum a posteriori (MAP) estimation problem because of its application in problems like computer vision and bioinformatics. On the other hand, researchers in the multi-agents community have studied the distributed constraint optimization problem (DCOP) because of its application in multi-agent coordination problems like sensor networks. However, both of these independently formulated problems bear strong similarities. For example, both problems are graphical models.

In this paper, we formally showed the similarities between these two problems and described three message passing algorithms, namely max-product linear programming (MPLP), expectation maximiation (EM) and max-sum (MS), that were developed to solve the MAP estimation problem. We also showed that these algorithms can be easily tailored to solve DCOPs as well. We demonstrated the feasibility of this approach with a preliminary set of experiments, where we evaluated MPLP on sensor network problems. The results showed that MPLP can find solutions with very small error bounds and finds them by several orders of magnitude faster than BnB-Adopt, an optimal DCOP search algorithm. Therefore, we believe that they demonstrate the potential of message passing algorithms to solve DCOPs.

Bibliography

[1] S. Aji, G. Horn, R. McEliece, and M. Xu. Iterative min-sum decoding of tail-biting codes. In *Information Theory Workshop, 1998*, pages 68–69, 1998.

[2] R. Dechter. *Constraint Processing*. Morgan Kaufmann, 2003.

[3] A. Farinelli, A. Rogers, A. Petcu, and N. Jennings. Decentralised coordination of low-power embedded devices using the Max-Sum algorithm. In *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 639–646, 2008.

[4] S. Fitzpatrick and L. Meertens. Distributed coordination through anarchic optimization. In V. Lesser, C. Ortiz, and M. Tambe, editors, *Distributed Sensor Networks: A Multiagent Perspective*, pages 257–295. Kluwer, 2003.

[5] A. Globerson and T. Jaakkola. Fixing Max-Product: Convergent message passing algorithms for MAP LP-relaxations. In *Advances in Neural Information Processing Systems (NIPS)*, pages 553–560, 2007.

[6] K. Hirayama and M. Yokoo. The distributed breakout algorithms. *Artificial Intelligence*, 161(1-2):89–115, 2005.

[7] J. K. Johnson, D. M. Malioutov, and A. S. Willsky. Lagrangian relaxation for map estimation in graphical models. In *Proceedings of the Allerton Conference on Communication, Control and Computing*, 2007.

[8] C. Kiekintveld, Z. Yin, A. Kumar, and M. Tambe. Asynchronous algorithms for approximate distributed constraint optimization with quality bounds. In *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 133–140, 2010.

[9] V. Kolmogorov. Convergent tree-reweighted message passing for energy minimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28:1568–1583, 2006.

[10] N. Komodakis, N. Paragios, and G. Tziritas. MRF optimization via dual decomposition: Message-passing revisited. In *IEEE International Conference on Computer Vision (ICCV)*, pages 1–8, 2007.

[11] A. Kumar, B. Faltings, and A. Petcu. Distributed constraint optimization with structured resource constraints. In *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 923–930, 2009.

[12] A. Kumar and S. Zilberstein. MAP estimation for graphical models by likelihood maximization. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1180–1188, 2010.

[13] R. Maheswaran, M. Tambe, E. Bowring, J. Pearce, and P. Varakantham. Taking DCOP to the real world: Efficient complete solutions for distributed event scheduling. In *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 310–317, 2004.

[14] R. Mailler and V. Lesser. Solving distributed constraint optimization problems using cooperative mediation. In *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 438–445, 2004.

[15] P. Modi, W.-M. Shen, M. Tambe, and M. Yokoo. ADOPT: Asynchronous distributed constraint optimization with quality guarantees. *Artificial Intelligence*, 161(1-2):149–180, 2005.

[16] J. Pearce and M. Tambe. Quality guarantees on k-optimal solutions for distributed constraint optimization problems. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1446–1451, 2007.

[17] J. Pearl. *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann, 1988.

[18] A. Petcu and B. Faltings. DPOP: A scalable method for multiagent constraint optimization. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 266–271, 2005.

[19] A. Petcu and B. Faltings. Superstabilizing, fault-containing multiagent combinatorial optimization. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, pages 449–454, 2005.

[20] A. Petcu and B. Faltings. Optimal solution stability in dynamic, distributed constraint optimization. In *Proceedings of the International Conference on Intelligent Agent Technology (IAT)*, pages 321–327, 2007.

[21] P. Ravikumar, A. Agarwal, and M. J. Wainwright. Message-passing for graph-structured linear programs: Proximal projections, convergence and rounding schemes. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 800–807, 2008.

[22] P. Ravikumar and J. Lafferty. Quadratic programming relaxations for metric labeling and Markov random field MAP estimation. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 737–744, 2006.

[23] D. Sontag, A. Globerson, and T. Jaakkola. Clusters and coarse partitions in LP relaxations. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1537–1544, 2008.

[24] D. Sontag, A. Globerson, and T. Jaakkola. Introduction to dual decomposition for inference. In S. Sra, S. Nowozin, and S. Wright, editors, *Optimization for Machine Learning*. MIT Press, 2010.

[25] D. Sontag, T. Meltzer, A. Globerson, T. Jaakkola, and Y. Weiss. Tightening LP relaxations for MAP using message passing. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 503–510, 2008.

[26] E. Sultanik, R. Lass, and W. Regli. Dynamic configuration of agent organizations. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 305–311, 2009.

[27] S. Ueda, A. Iwasaki, and M. Yokoo. Coalition structure generation based on distributed constraint optimization. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pages 197–203, 2010.

[28] M. Vinyals, M. Pujol, J. Rodriguez-Aguilarhas, and J. Cerquides. Divide-and-coordinate: DCOPs by agreement. In *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 149–156, 2010.

[29] M. Vinyals, E. Shieh, J. Cerquides, J. Rodríguez-Aguilar, Z. Yin, M. Tambe, and E. Bowring. Quality guarantees for region optimal DCOP algorithms. In *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2011.

[30] M. Wainwright, T. Jaakkola, and A. Willsky. MAP estimation via agreement on (hyper)trees: Message-passing and linear programming approaches. *IEEE Transactions on Information Theory*, 51:3697–3717, 2002.

[31] M. Wainwright, T. Jaakkola, and A. Willsky. Tree consistency and bounds on the performance of the max-product algorithm and its generalizations. *Statistics and Computing*, 14:143–166, 2004.

[32] M. J. Wainwright and M. I. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1:1–305, 2008.

[33] T. Werner. A linear programming approach to max-sum problem: A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29:1165–1179, 2007.

[34] T. Werner. What is decreased by the max-sum arc consistency algorithm? In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 1007–1014, 2007.

[35] C. Yanover, T. Meltzer, and Y. Weiss. Linear programming relaxations and belief propagation – an empirical study. *Journal of Machine Learning Research*, 7:2006, 2006.

[36] W. Yeoh, A. Felner, and S. Koenig. BnB-ADOPT: An asynchronous branch-and-bound DCOP algorithm. *Journal of Artificial Intelligence Research*, 38:85–133, 2010.

[37] W. Yeoh, P. Varakantham, X. Sun, and S. Koenig. Incremental DCOP search algorithms for solving dynamic DCOPs (Extended Abstract). In *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2011.

[38] W. Zhang, G. Wang, Z. Xing, and L. Wittenberg. Distributed stochastic search and distributed breakout: Properties, comparison and applications to constraint optimization problems in sensor networks. *Artificial Intelligence*, 161(1-2):55–87, 2005.

[39] R. Zivan. Anytime local search for distributed constraint optimization. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pages 393–398, 2008.